DIGITAL SIGNAL PROCESSING LABORATORY

EXPERIMENT-1 - SAMPLING

Group:- 32

Authors:-

Jaya Kishnani (20EC30020)

Gunjan Shekhar(20EC10032)

TA:- Abhishek Singh

Aim:

a) Sampling of a sinusoidal waveform
b) Sampling below Nyquist rate and effect of aliasing
c) Spectrum of a square wave
d) Interpolation or upsampling of a signal

Theory:

<u>Sampling</u>-

The Nyquist Sampling Theorem says that a band-limited signal can be perfectly reconstructed from its samples provided the sampling frequency is greater than twice its highest frequency component.

<u>Nyquist sampling rate</u>-

For a band-limited signal with a limit at B Hz, Nyquist sampling rate should be greater than 2B Hz to avoid aliasing and for proper reconstruction of the signal.

<u>Interpolation</u>-

If an analog signal is sampled at a frequency higher than Nyquist frequency, say Fs_1, we can interpolate the intermediate (L-1) samples and obtain the samples at a frequency $Fs_2 = LFs_1$.

<u>Part A</u>

Observation:

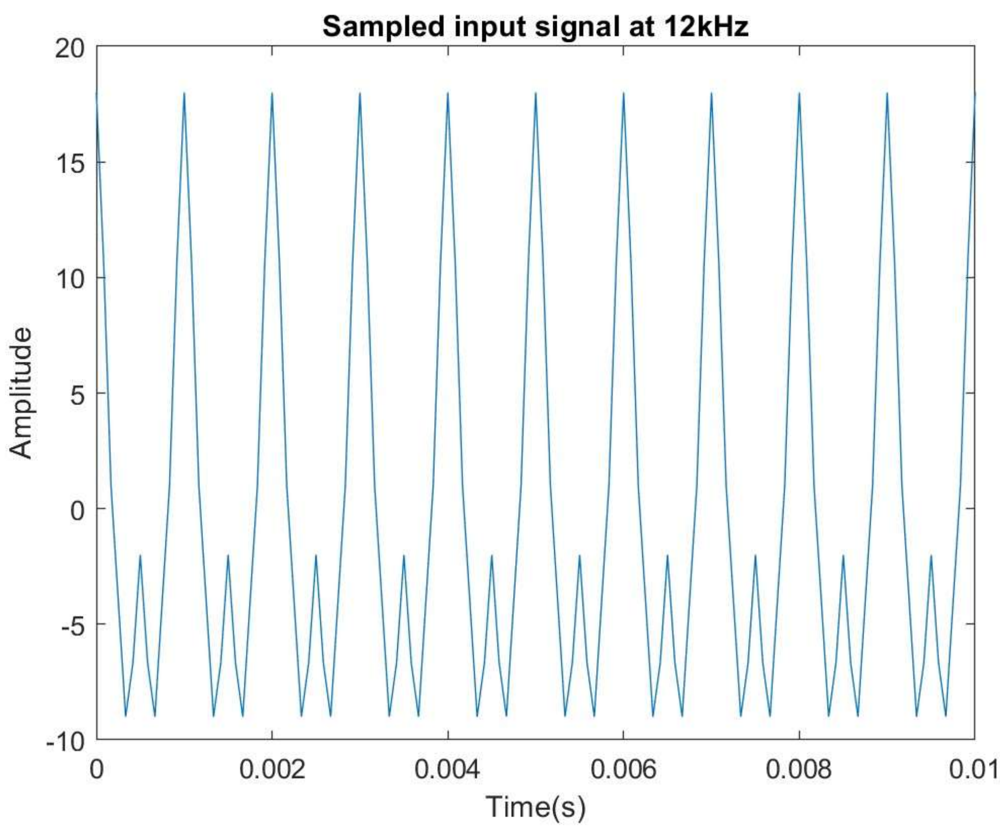
Code

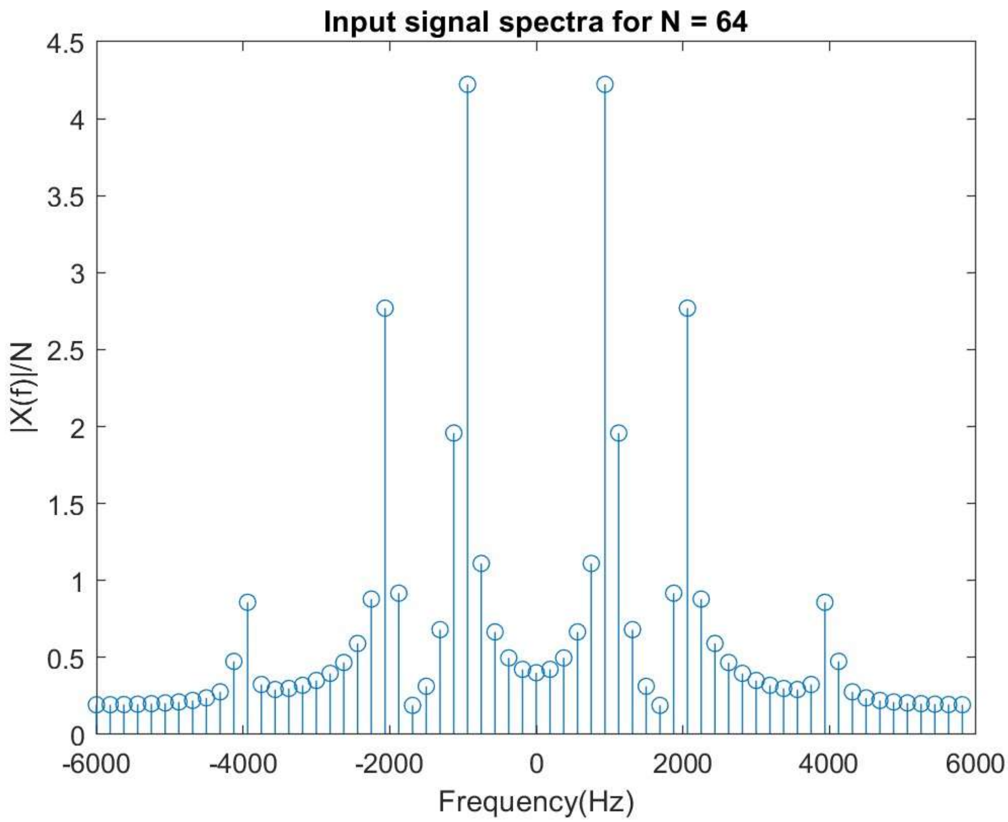
```
% Given parameters

Fs = 12000; %Sampling frequency

T = 0.2;

t = 0:1/Fs:T;

f = 1000;

%Given signal

x = 10*cos(2*pi*f*t)+6*cos(2*pi*2*f*t)+2*cos(2*pi*4*f*t);

%Plot
```

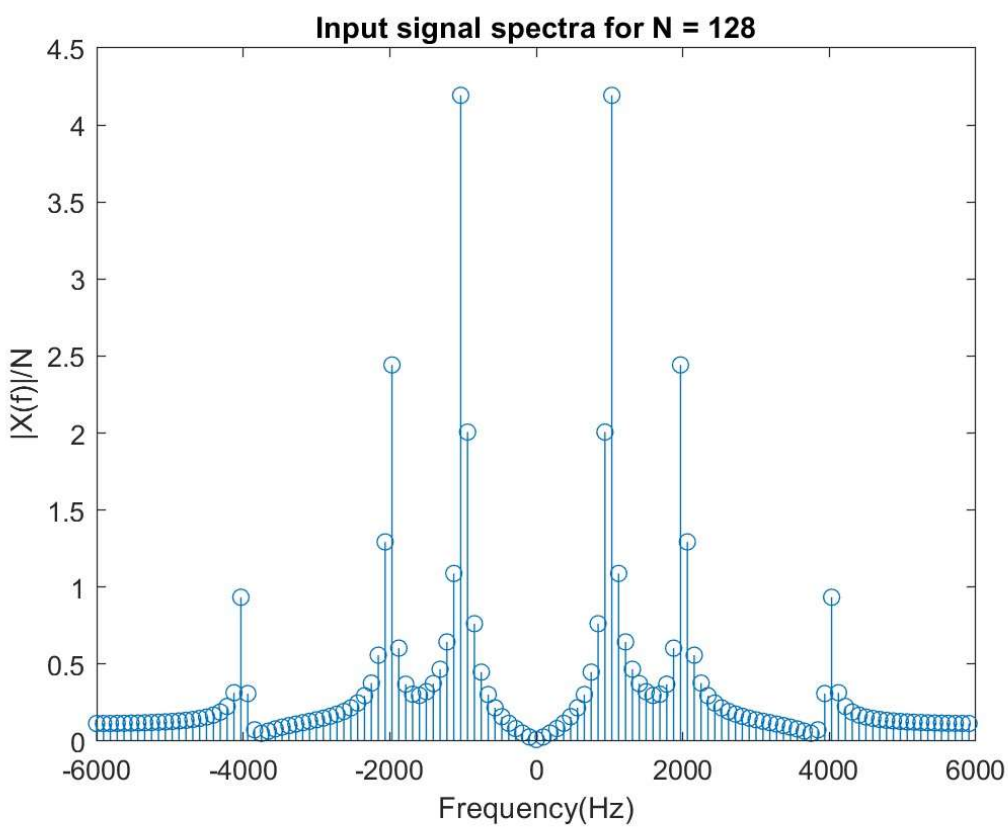
```
figure(1)
plot(t, x)
title('Sampled input signal at 12kHz')
xlabel('Time(s)')
ylabel('Amplitude')
xlim([0,0.01])
%Signal spectra
%For N=64
N=64;
f=Fs/N*(-N/2:N/2-1);
X = fft(x, N);
X = fftshift(X);
X = abs(X);
figure(2)
stem(f, X/N)
title('Input signal spectra for N = 64')
xlabel('Frequency(Hz)')
ylabel('|X(f)|/N')
%For N=128
N=128;
f=Fs/N*(-N/2:N/2-1);
X = fft(x, N);
X = fftshift(X);
X = abs(X);
figure(3)
stem(f, X/N)
title('Input signal spectra for N = 128')
xlabel('Frequency(Hz)')
ylabel('|X(f)|/N')
%For N=256
```

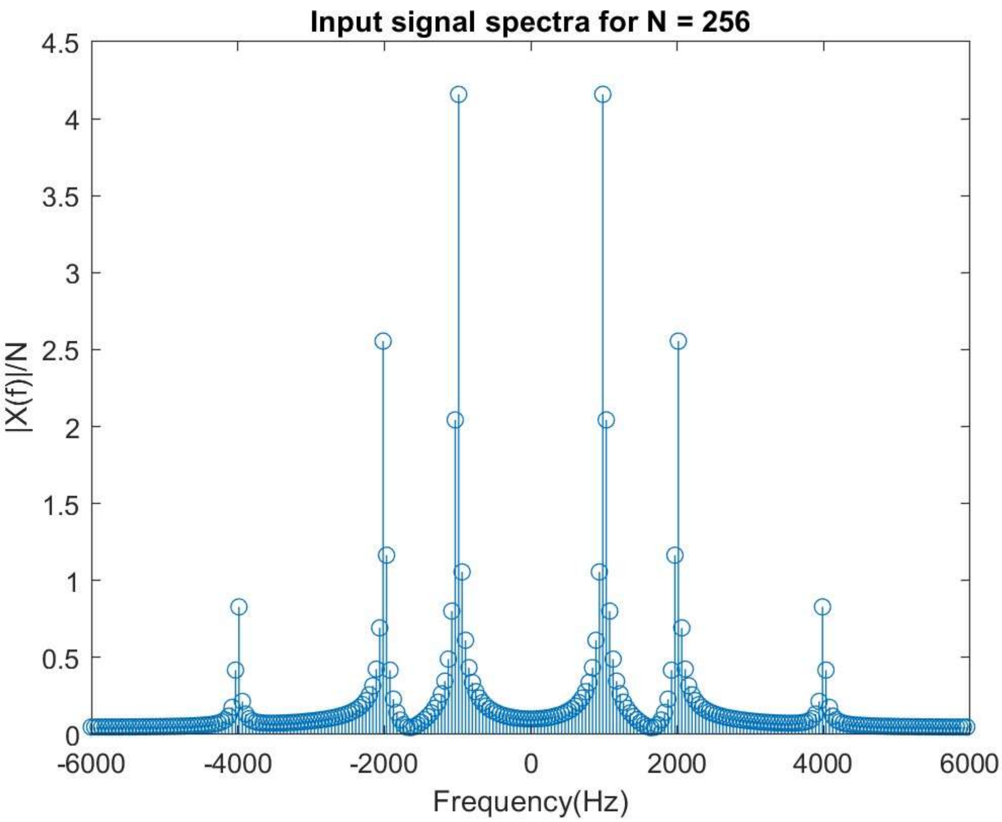
```
N=256;

f=Fs/N*(-N/2:N/2-1);

X = fft(x, N);

X = fftshift(X);

X = abs(X);

figure(4)

stem(f, X/N)

title('Input signal spectra for N = 256')

xlabel('Frequency(Hz)')

ylabel('|X(f)|/N')
```

Images-




Input signal spectra for N = 64
|X(f)|/N
Frequency(Hz)
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
-6000
-4000
-2000
0
2000
4000
6000


Input signal spectra for N = 128
|X(f)|/N
Frequency(Hz)
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
-6000
-4000
-2000
0
2000
4000
6000



Discussion:

A higher value of N results in a higher number of samples of the frequency domain representation of the sampled time domain signal. We observe that the plot of the N-point DFT has less spread near the frequencies in the signal and is more precise and smooth as the value of N is increased.

Part B

Observation:

Code

```
% Nyguist rate and aliasing effect

T=50;

N=256;

F = 1000;

Fs = 8000; %Sampling frequency

t = 0:1/Fs:T;
```

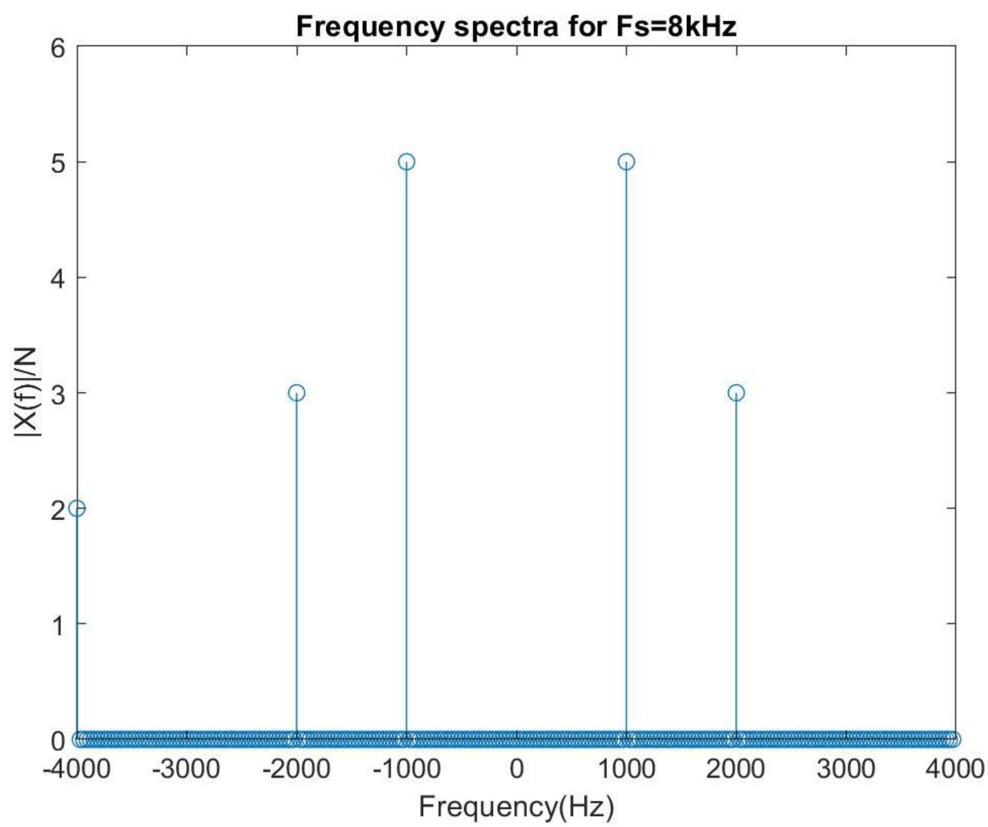
```
x = 10*cos(2*pi*F*t) + 6*cos(2*2*pi*F*t) + 2*cos(2*4*pi*F*t); %input signal
Y = fft(x,N); %Fast fourier transform
Y = fftshift(Y);% Shifting FFT to center
Y = abs(Y);
Y = Y/N;
f = -1*(Fs/2):Fs/N:(Fs/2)-1;
figure(1);
stem(f,Y)  %fs = 2*fm => aliasing effect
```

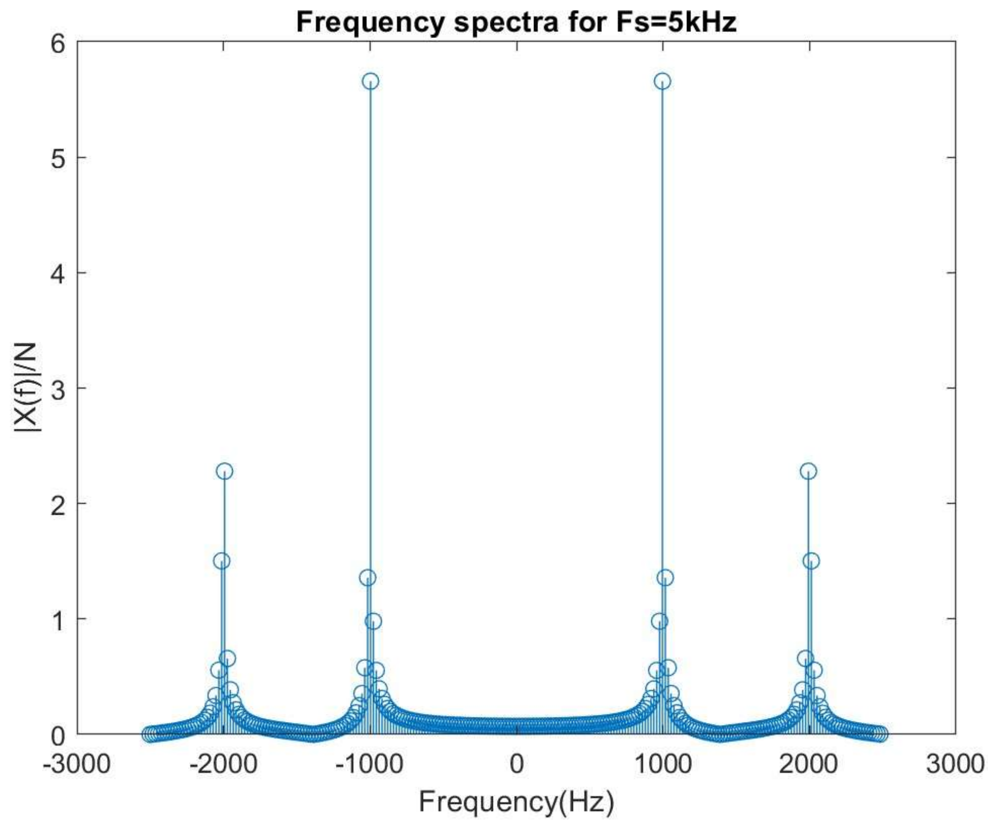
```
T=50;
N=256;
F = 1000;
Fs = 5000;
t = 0:1/Fs:T;
x = 10*cos(2*pi*F*t) + 6*cos(2*2*pi*F*t) + 2*cos(2*4*pi*F*t);
Y = fft(x,N);
Y = fftshift(Y);
Y = abs(Y);
Y = Y/N;
f = -1*(Fs/2):Fs/N:(Fs/2)-1;
figure(2);
stem(f,Y) % fs<2*fm => aliasing effect
```

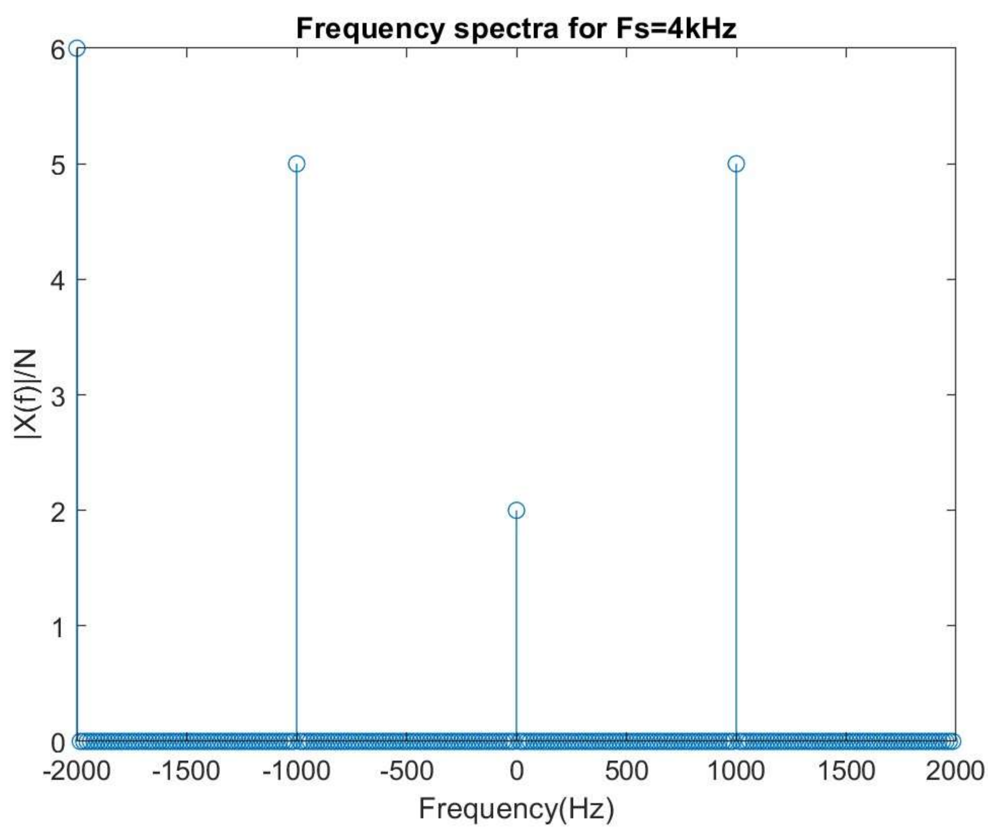
```
T=50;
N=256;
F = 1000;
Fs = 4000;
t = 0:1/Fs:T;
```

```
x = 10*cos(2*pi*F*t) + 6*cos(2*2*pi*F*t) + 2*cos(2*4*pi*F*t);
Y = fft(x,N);
Y = fftshift(Y);
Y = abs(Y);
Y = Y/N;
f = -1*(Fs/2):Fs/N:(Fs/2)-1;
figure(3);
stem(f,Y) % fs<2*fm => aliasing effect
```

Images-



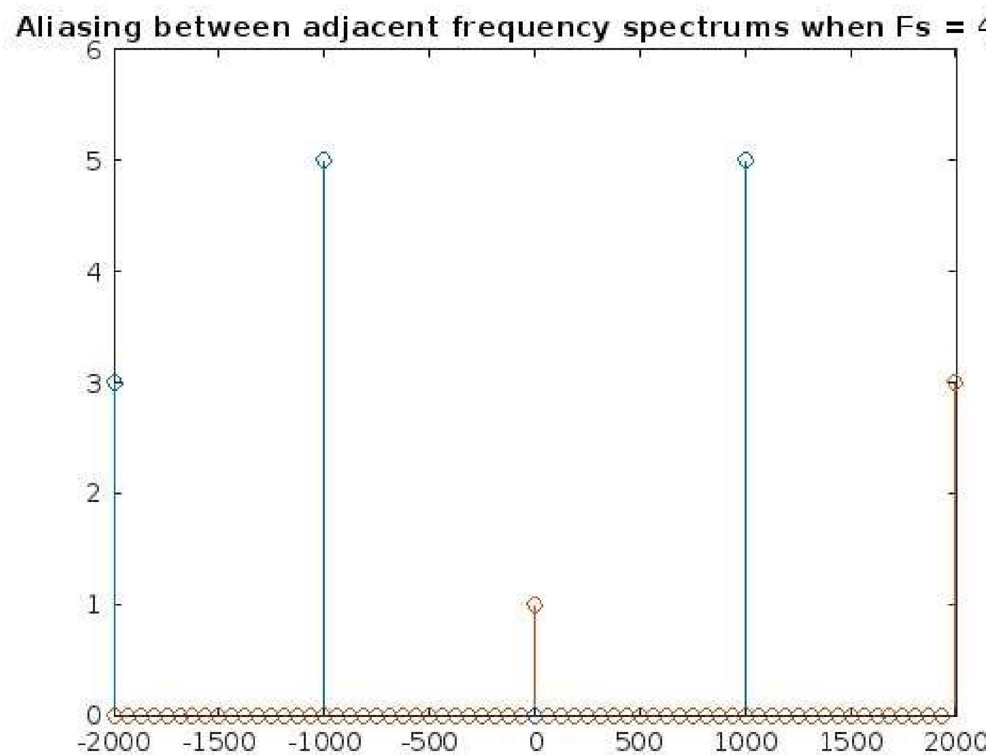

Frequency spectra for Fs=5kHz
|X(f)|/N
Frequency(Hz)
0
1
2
3
4
5
6
-3000
-2000
-1000
0
1000
2000
3000

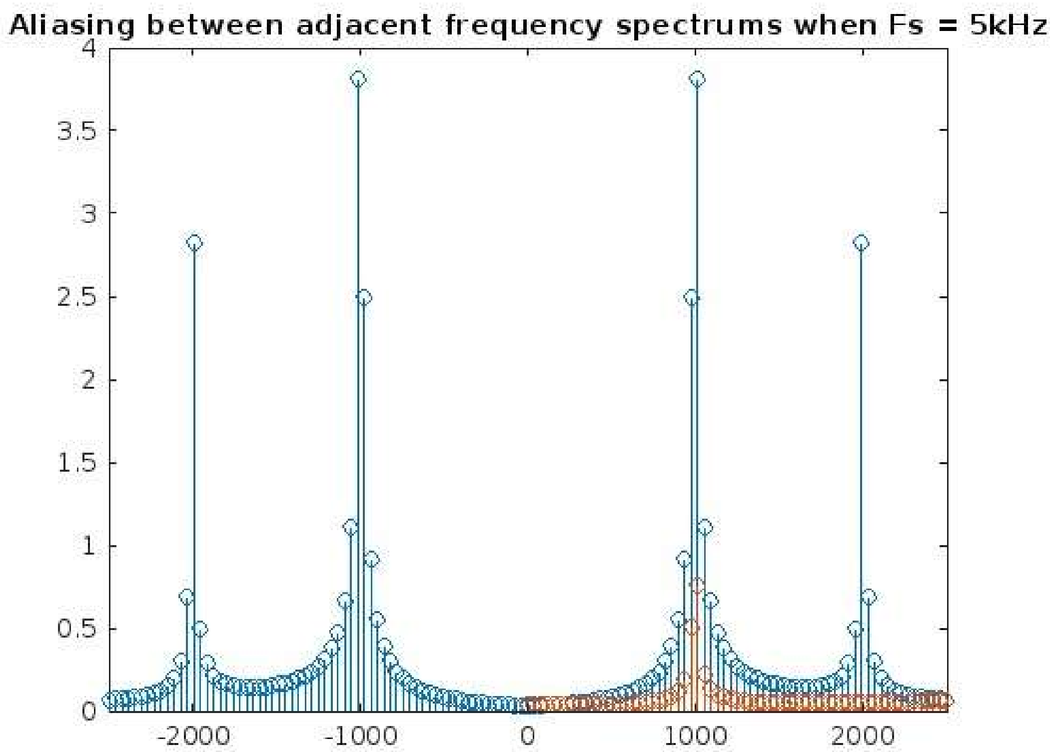

Frequency spectra for Fs=4kHz
|X(f)|/N
Frequency(Hz)
0
1
2
3
4
5
6
-2000
-1500
-1000
-500
0
500
1000
1500
2000

Discussion:

We get the required N-point DFT of the signal x(t) when we set the sampling frequency to 8 kHz. The plot has non-zero value only at the frequencies: -4 kHz, -2 kHz, -1 kHz, 1 kHz and 2 kHz. The graph is aliased at the points -4 kHz and 4 kHz which can also be concluded by observing the amplitudes of the DFT at those frequencies. This is because our sampling frequency is not greater than twice the maximum frequency component. Therefore the inequality in the Nyquist sampling theorem should be strictly followed.

When the sampling frequency is below 8 kHz (i.e., set to 5 kHz and 4kHz), we observe that aliasing has occurred and information is lost. The frequency spectrum of the sampled signal is a superposition of infinite frequency spectrums of continuous time signal x(t) each centred at multiples of sampling frequency. In the plots below the adjacent frequency spectrums of the continuous time signal x(t) coalesce with each other because the sampling frequency is less and there's overlapping between the spectrums.





Part C

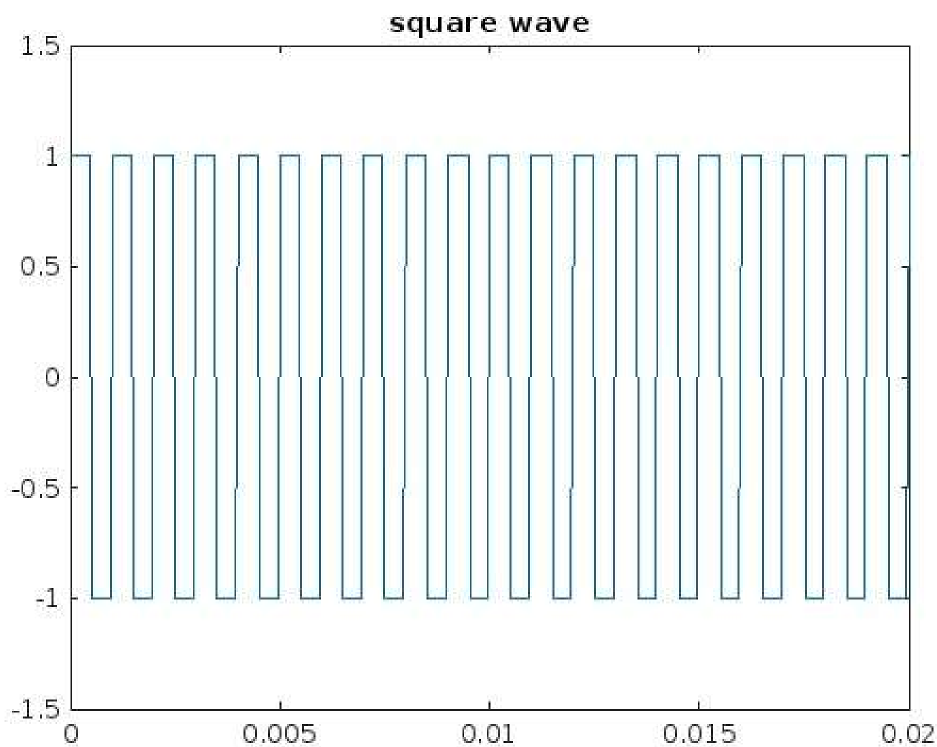
Observation:

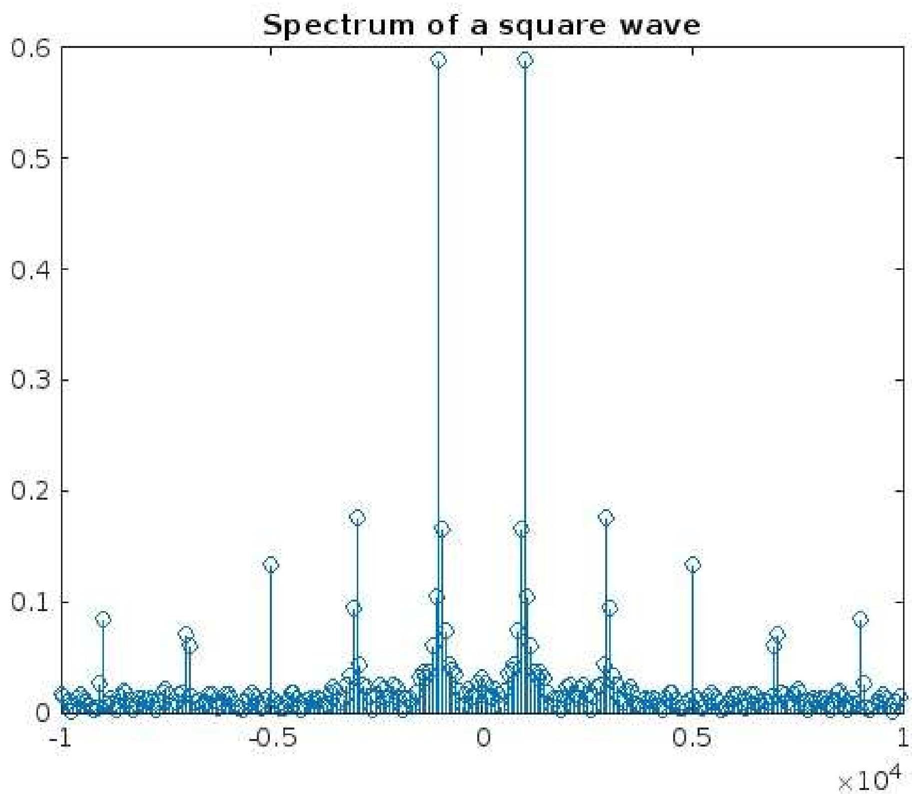
Code

```
T=0.02;

N=256;

F = 1000;

Fs = 20000; %sampling frequency

t = 0:1/Fs:T;

x = square(2*pi*F*t); %Square wave

figure(1);

plot(t,x)

title('square wave')

ylim([-1.5, 1.5])

Y = fft(x,N); %Fast fourier transform
```

```
Y = fftshift(Y); %Shifting FFT to center

Y = abs(Y);

Y = Y/N;

f = -1*(Fs/2):Fs/N:(Fs/2)-1;

figure(2);

stem(f,Y) %Plotting spectrum of the given square wave @discrete

title('Spectrum of a square wave')
```

Images:-





Discussion:

The fourier series representation of a square wave is-

x(t) =$\frac{1}{2} + \frac{2}{\Pi}\sum_{n=1}^{\infty}\frac{sin(2\Pi f(2n-1)t)}{2n-1}$

The frequencies in the fourier series decomposition of the square wave consists of only the odd multiples of the frequency of the square wave (F). This can also be seen in the DFT plot where the magnitude is highest near the odd multiples of F.

Part D

Observation:

Code

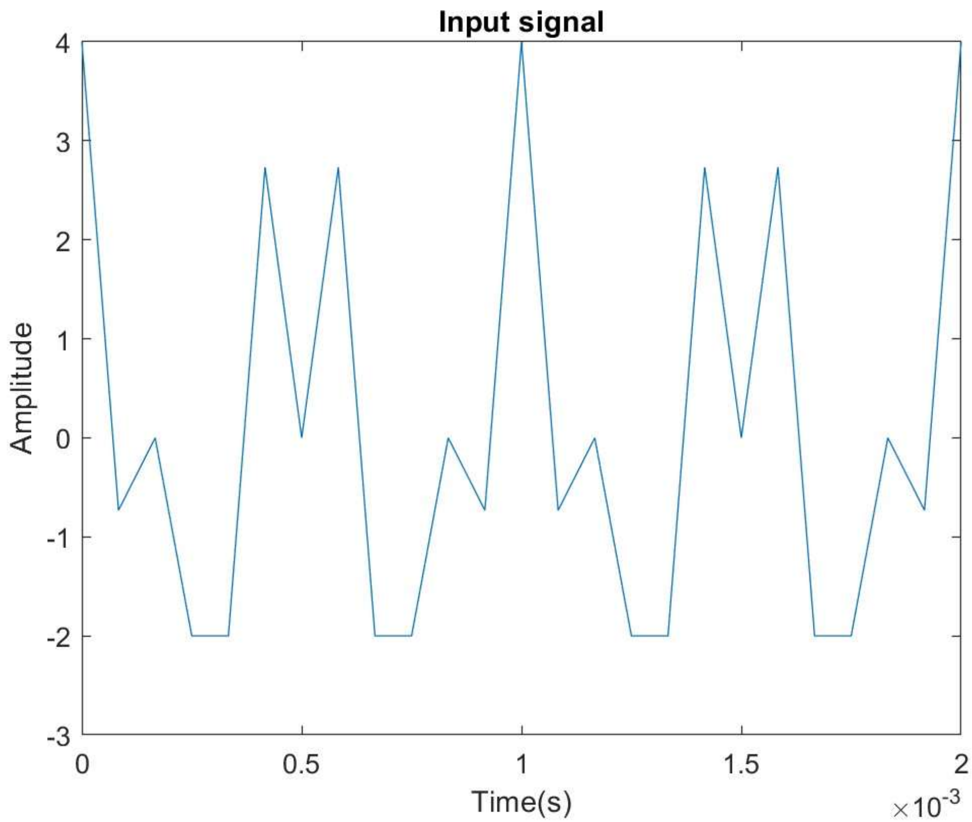
```
%input parameters

T=0.002;

N=256;

F = 1000;
```

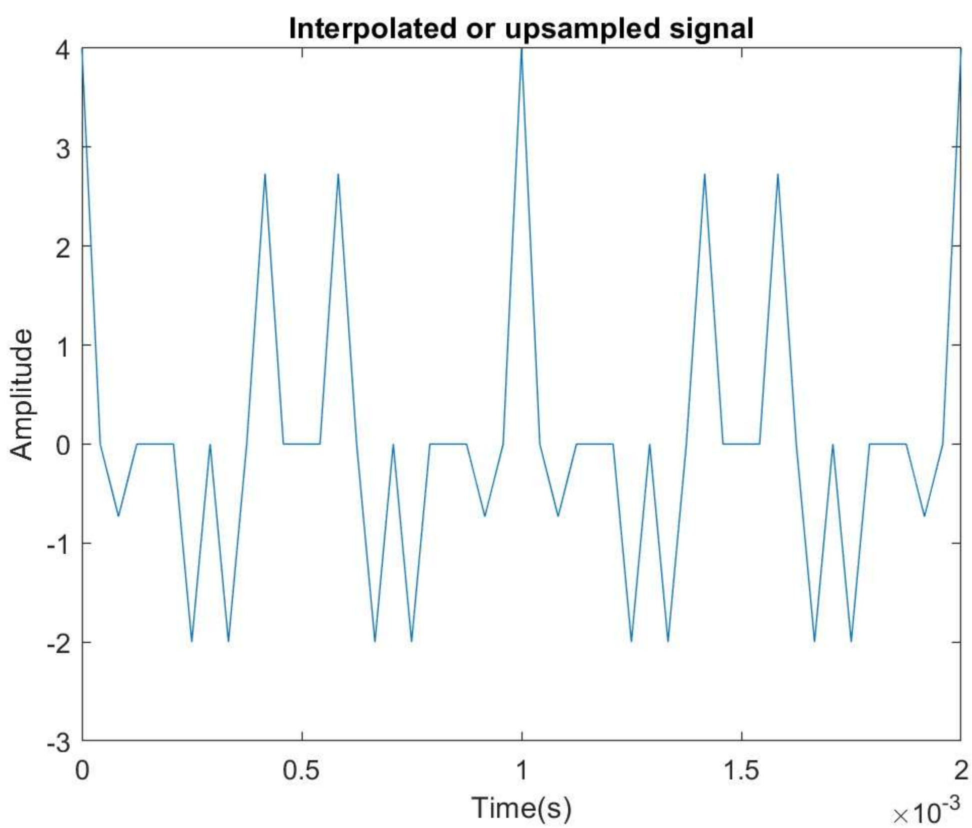
```
Fs1 = 12000; %Sampling frequency

Fs2 = 24000;

Fc = 6000;  %cut-off frequency

t = 0:1/Fs1:T;

x =2*cos(2*5*pi*F*t)+2*cos(2*2*pi*F*t); %low pass input signal of
bandwidth 6kHz

figure(1)

plot(t,x) %Plotting input signal

title('Input signal')

xlabel('Time(s)')

ylabel('Amplitude')

x_1 = upsample(x,2); %upsampling

t_1 = 0:(1/(2*Fs1)):T+ (1/(2*Fs1));

figure(2)

plot(t_1,x_1)

title('Interpolated or upsampled signal')%adding zeros distort the
signal

xlabel('Time(s)')

ylabel('Amplitude')

%Filters

%Butterworth filter - flat magnitude filter

[b,a] = butter(6, Fc/(Fs2/2)); % 6th-order lowpass digital
Butterworth

dataout = filter(b,a,x_1);

figure(3)

plot(t_1, dataout)

title('butterworth filter output')% obtained attenuated signal

xlabel('Time(s)')

ylabel('Amplitude')

%Chebyshev1 Filter
```

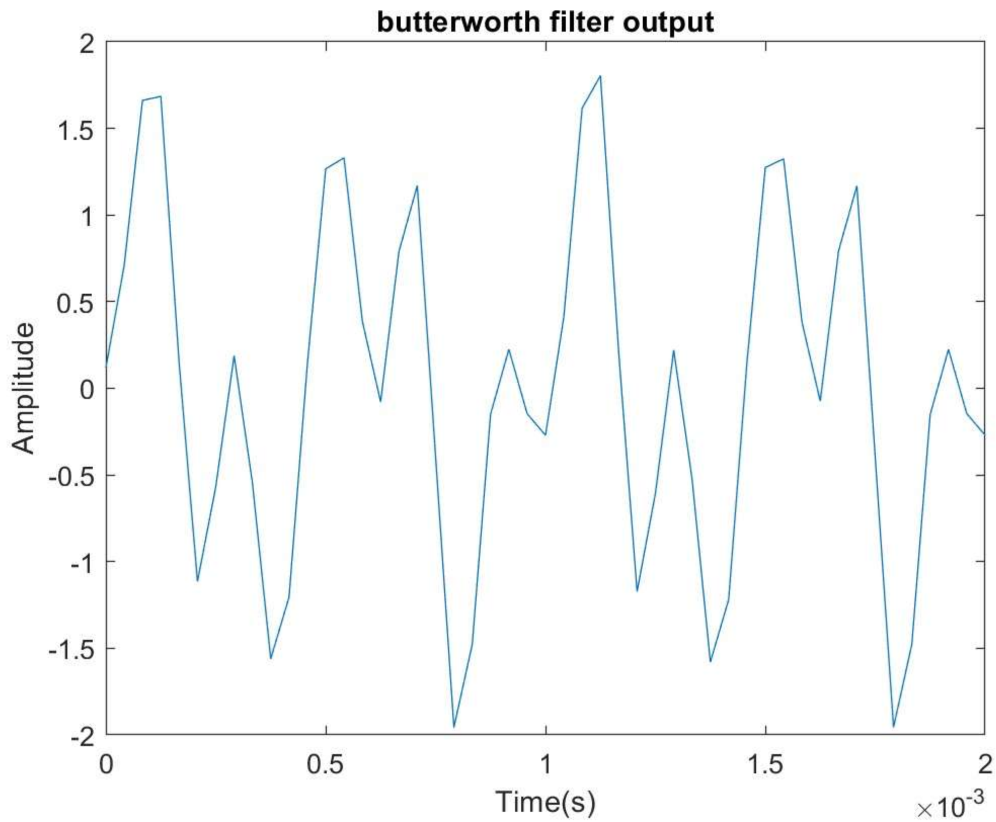
```
[b,a] = cheby1(6,5, Fc/(Fs2/2)); %6th-order lowpass digital Chebyshev
Type I filter with normalized passband edge frequency

output_cheby1= filter(b,a,x_1);

figure(4)

plot(t_1, output_cheby1)

title('Chebyshev1 filter output')% obtained attenuated signal

xlabel('Time(s)')

ylabel('Amplitude')

%Chebyshev2 filter

[b,a] = cheby2(6,2, Fc/(Fs2/2)); %6th-order lowpass digital Chebyshev
Type II filter with normalized stopband edge frequency

output_cheby2= filter(b,a,x_1);

figure(5)

plot(t_1, output_cheby2)

title('Chebyshev2 filter output')% obtained attenuated signal

xlabel('Time(s)')

ylabel('Amplitude')

%Elliptic filter

[b,a] = ellip(6,10,50,Fc/(Fs2/2)); % 6th-order lowpass digital
elliptic filter with normalized passband edge frequency

output_ellip= filter(b,a,x_1);

figure(6)

plot(t_1, output_ellip)

title('Elliptic filter output')% obtained attenuated signal

xlabel('Time(s)')

ylabel('Amplitude')

%FIR1

b=fir1(6,[0.0001,0.5]);%Hamming window used, 6th-order lowpass, with
linear phase

fir1_output = filter(b,1, x_1);

figure(7)
```

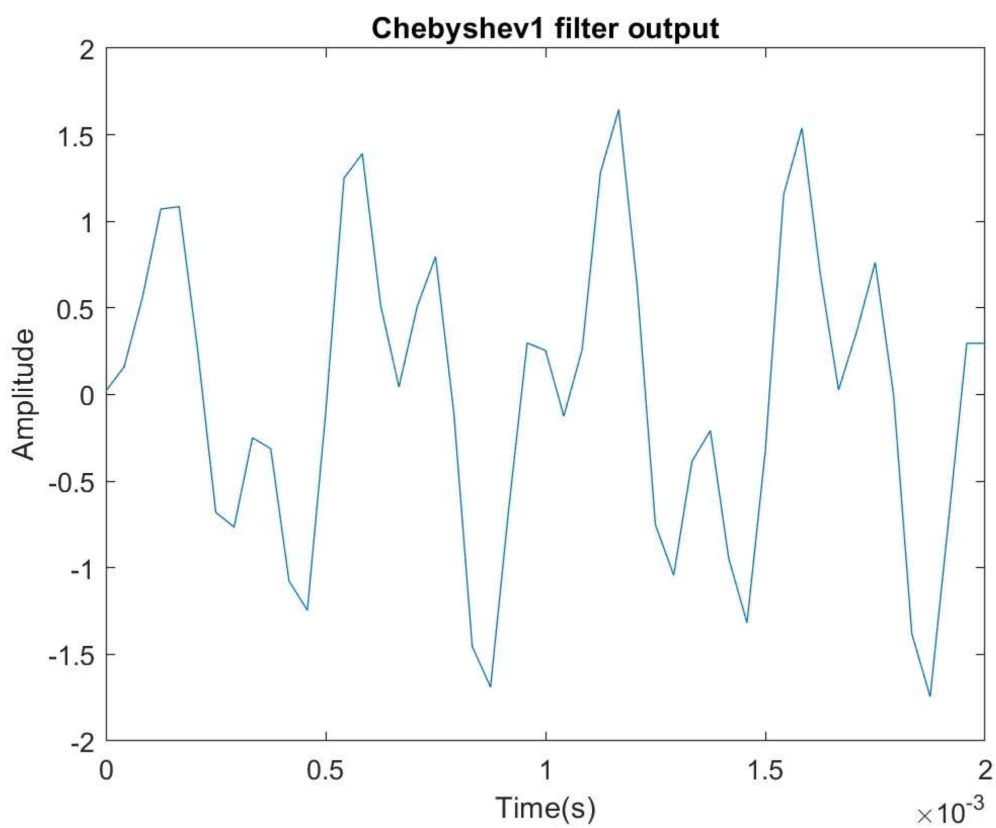
```
plot(t_1, fir1_output)

title('FIR1 filter output')% obtained attenuated signal

xlabel('Time(s)')

ylabel('Amplitude')

%FIR2

f=[0 0.5 0.5 1];%frequency

m=[1 1 0 0]; %magnitude

b=fir2(6,f,m); %6th-order FIR filter with frequency-magnitude
characteristics

fir2_output=filter(b,1,x_1);

figure(8)

plot(t_1, fir2_output)

title('FIR2 filter output')% obtained attenuated signal

xlabel('Time(s)')

ylabel('Amplitude')
```

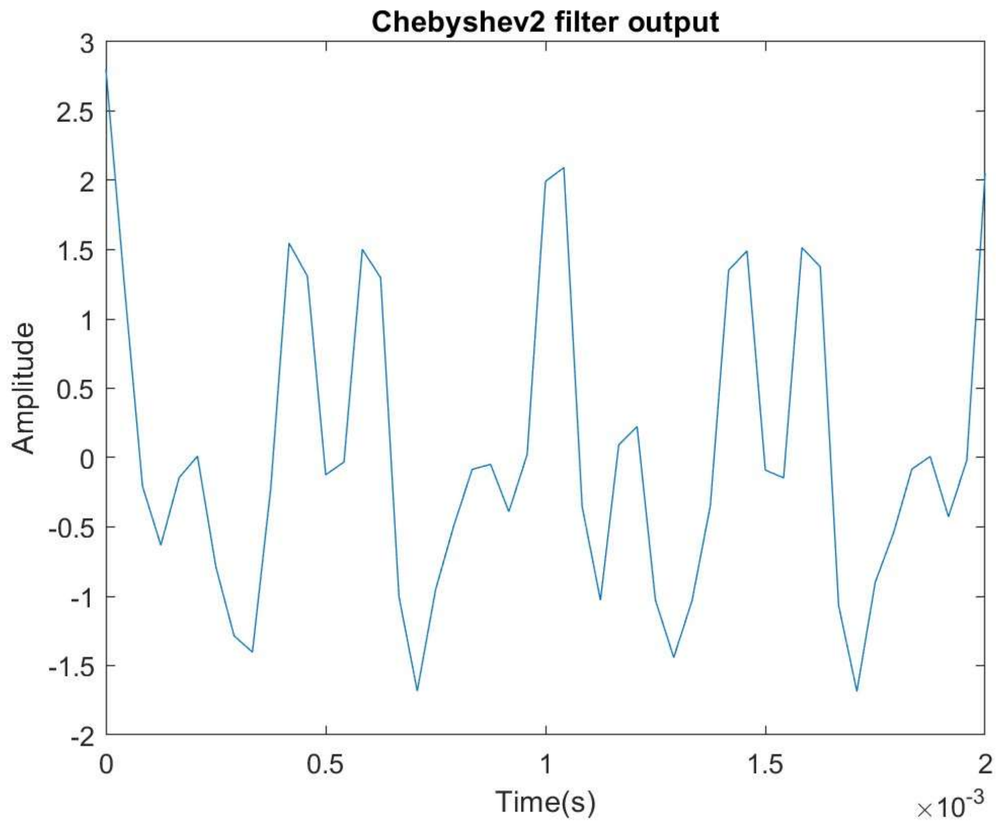
Images:-

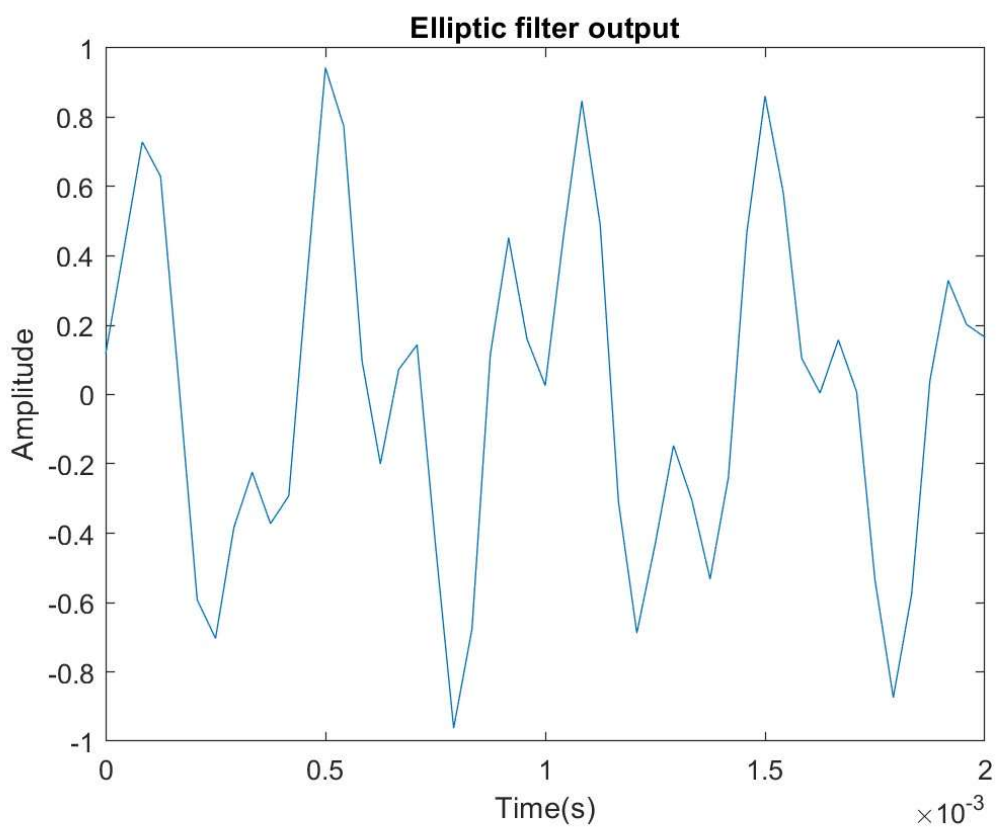

Input signal
Amplitude
Time(s)
$\times 10^{-3}$
4
3
2
1
0
-1
-2
-3
0
0.5
1
1.5
2

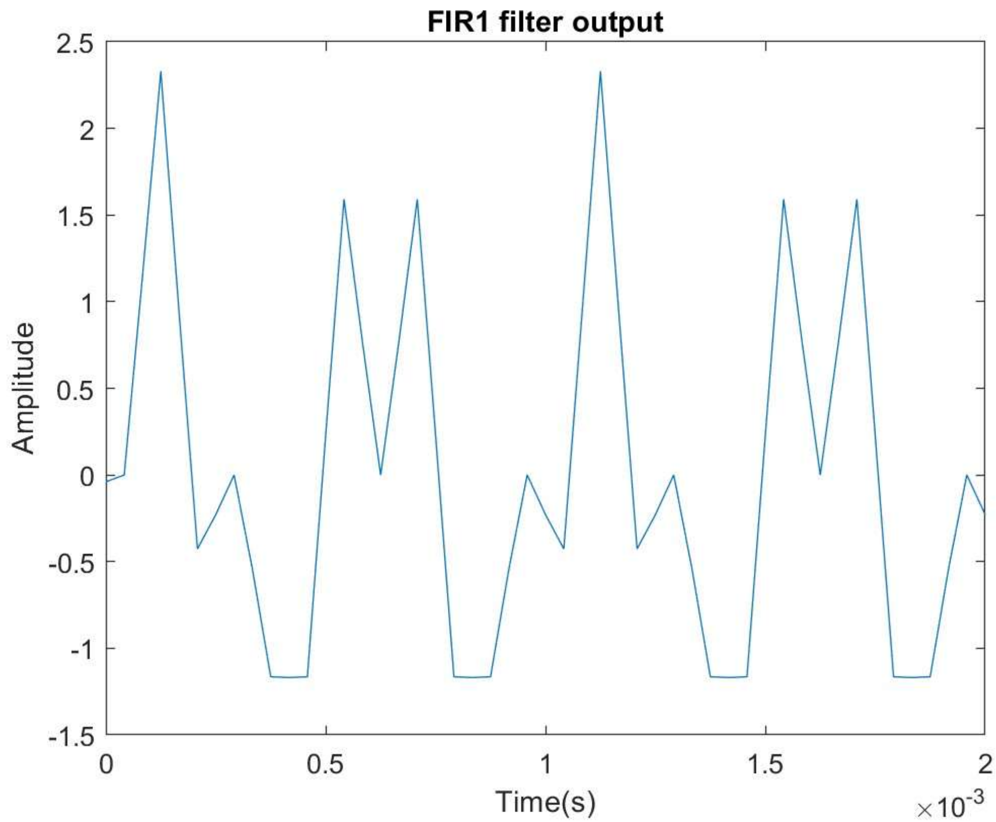

Interpolated or upsampled signal
Amplitude
Time(s)
$\times 10^{-3}$
4
3
2
1
0
-1
-2
-3
0
0.5
1
1.5
2

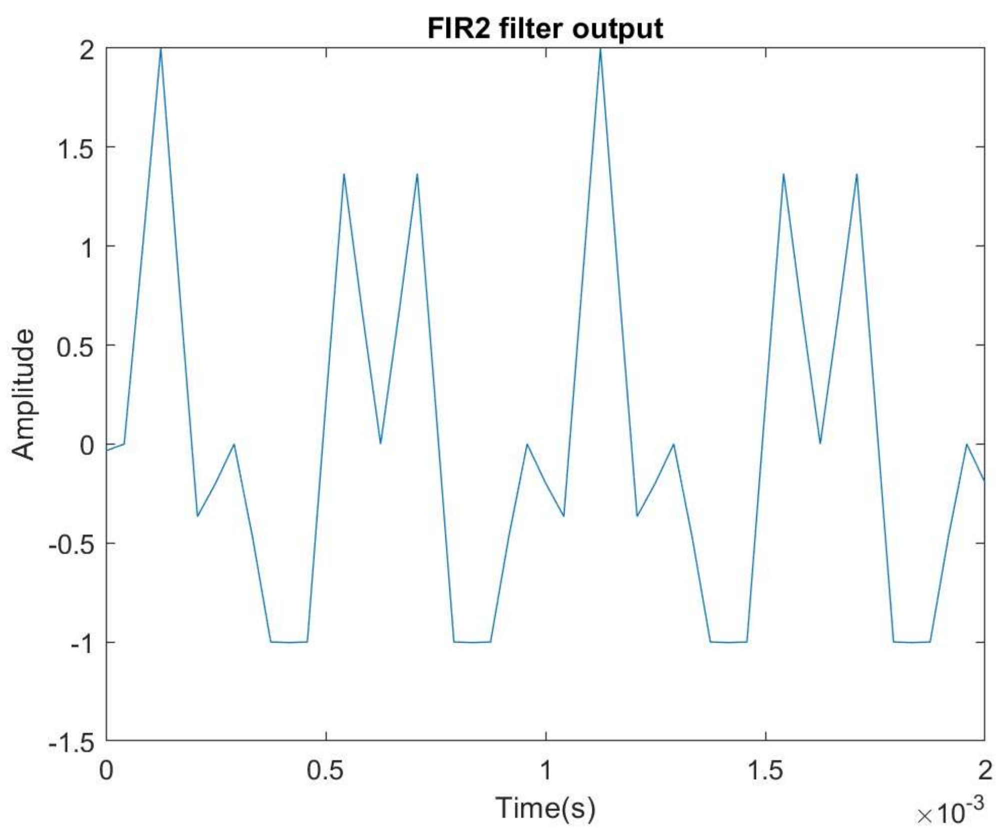

butterworth filter output
Amplitude
Time(s)
×10^-3
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2


Chebyshev1 filter output
Amplitude
Time(s)
×10^-3
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2


Chebyshev2 filter output
Amplitude
Time(s)
×10^-3
3
2.5
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
0
0.5
1
1.5
2


Elliptic filter output
Amplitude
Time(s)
×10^-3
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
0.5
1
1.5
2


FIR1 filter output
Amplitude
Time(s)
×10^-3
2.5
2
1.5
1
0.5
0
-0.5
-1
-1.5
0
0.5
1
1.5
2


FIR2 filter output
Amplitude
Time(s)
×10^-3
2
1.5
1
0.5
0
-0.5
-1
-1.5
0
0.5
1
1.5
2

Discussion:

We observe that after interpolation the shape of the signal gets slightly distorted. This is because of the insertion of zeros in between the samples so the signal is going to zero at each alternate sample thereby disturbing the shape of the wave.

The output signal obtained after passing through the low pass filter is less distorted as compared to the interpolated signal and has double the sampling frequency as compared to the original sampled signal (before interpolation).

The type of low pass filter used should be chosen correctly because it dictates the various characteristics of the obtained output signal. The Butterworth, Chebyshev and Elliptic filters seem to attenuate and distort the output signal much more than the FIR filters.